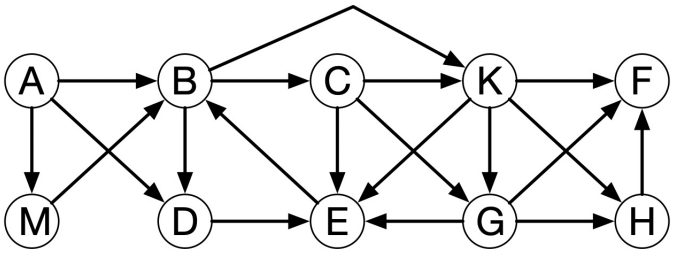
Assignment 5
CS-GY 6033 Fall 2024
Due Date: Dec 9th 11:55pm on Gradescope

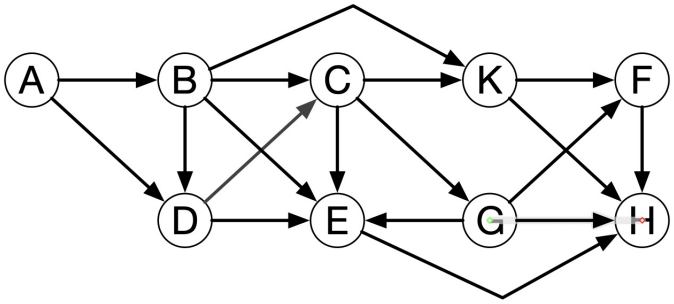
Question 1: Graph Traversals, Warm up

.

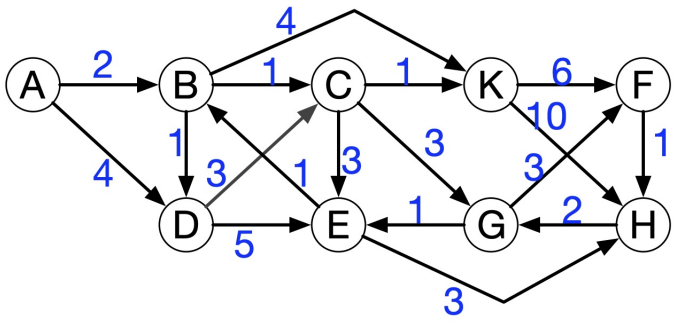
(a) 5 points Execute DFS on the directed graph shown below, using DFS-visit which keeps track of time stamps. Start with DFS-visit from vertex C. Upon completion, you must draw the resulting DFS tree(s) and label the edge types (as tee, forward, back, cross).



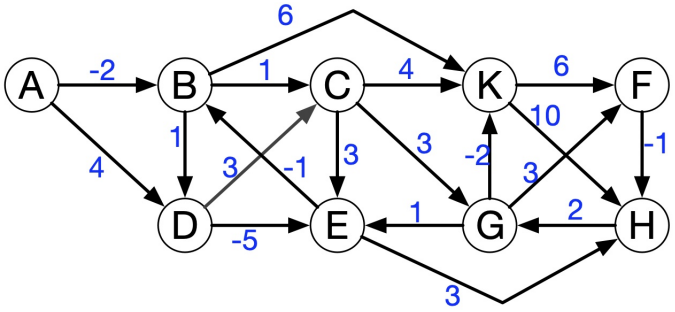
(b) 5 points Execute the algorithm from class for finding a topological sort on the directed acyclic graph shown below. Start DFS on vertex K and then A. You must indicate the start/finish times for each vertex and draw the final topo sort with edges shown.



(c) 5 points Execute Dijkstra's algorithm on the following directed weighted graph, using vertex A as the source You must show the values of $v.d$ and how they change as each edge is added.



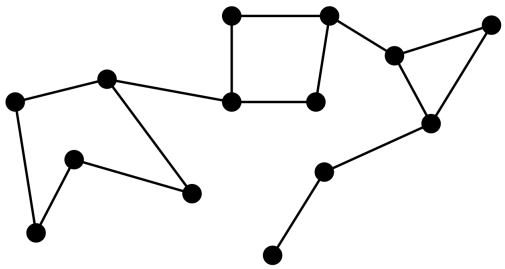
(d) 8 points Execute Bellman-Ford's algorithm on the following directed, weighted graph, using source vertex A. You must process the edges in lexicographic order (AB, AC, etc). You must show the values of $v.d$ and how they change as each edge is added. You must also show the current edges of the tree as the algorithm executes.



Question 2: Graph traversals on unweighted graphs

.

(a) 8 points Let G be a connected graph which may contain cycles, however all cycles are vertex-disjoint. This means that any two cycles in the graph **do not have a vertex in common.** An example of such a graph is shown below. Your job is to write a procedure called MaxCycle(v) which returns the maximum-length of a cycle in G. The input parameter v is any vertex in tehe. graph. Be sure the carefully describe any initialisation that is necessary before calling the procedure, and justify the runtime of $O(V + E)$.



(b) 8 points Let G be an undirected, connected graph. Update the pseudo-code for DFS-visit so that it returns the NUMBER of degree-one vertices in G. Call your algorithm CountLeaves(u) which takes as input any node u of the connected graph G. **Ensure that you do not use any new external variables.**. Instead use recursion to RETURN the correct result.

(c) 8 points A village on the island of Naxos in Greece is made up of n tiny farms. The old roads connecting the farms are so narrow, that they have all been designated as one-way roads. The island residents would like to ensure that they are able to drive to and from every pair of farms in the village, using a sequence of the roads. A brilliant computer scientist has arrived on the island, and modelled the island map using a directed graph: each farm is a vertex, and each one-way road is a directed edge. She then uses the strongly connected component algorithm on G to determine the number of strongly connected components. Unfortunately, she found that there are *two* SCCs, meaning some villages cannot be reached from other villages!

Your job: Design an algorithm that determines where to add **exactly one more road** so that it is possible to drive to and from every pair of villages. You do not need pseudo-code, but you must carefully describe your steps. You must determine the pair of cities that must be connected with the new road (don't concern yourself with intersections, etc). Justify the overall runtime must be $O(n^2)$.

(d) 8 points Let G be an undirected, connected graph with vertex set V and edge set E. The graph represents a hiking map where each vertex is a marker on the map, and each edge (u, v) is either a hiking trail between u and v or a **bridge** between markers u and v. Let $b(u, v)$ be an edge attribute that is *true* if the edge $e = (u, v)$ is a bridge, and false otherwise. The goal is to determine if there is a route from marker S to marker T that traverses *at most one bridge*. You must design an algorithm to solve this problem, which must be based on DFS, with a runtime of $O(V + E)$. The procedure must return *true* if there is a route and *false* otherwise.

(e) 6 points Suppose the above algorithm returns true. Your job is to write a procedure that outputs the routs from S to T. The output consists of a sequence of print statements, where you print out each of the trail makers. The markers must come out in the correct order: from S to T.

Question 3: Spanning Trees

(a) 6 points Let G be a weighted, undirected graph on vertex set V and edge set E. Suppose we run Kruskal's algorithm, and store the edges of the MST in a list of edges, T. However, once the edges have been stored in T, an error occurs, and an additional edge from E is **accidentally added to the set** T. This means that the edges in T no longer represents the edges of the MST. Your job is to design an algorithm that restores T so that it contains the edges of the MST. Your algorithm must run in time $O(V)$, and therefore cannot simply re-compute the MST from scratch, which would take $O(ElogV)$. You do not need to write the pseudo-code, but you must carefully describe all aspects of your algorithm, including how you design any input, and describe the steps of the procedure that determines which edge of T can be removed. Justify the runtime of each stop.

(b) 4 points Suppose a weighted graph G has distinct edges weights. Is there more than one possible MST ? Justify your answer.

(c) 4 points Draw a graph on 10 vertices and 15 edges where the MST produces the same tree as the SSSP from a particular vertex.

Question 4: Weighted graphs

.

(a) 10 points Consider a road map which has n marked intersections and m roads. Each road is a bi-directional connection between two intersections on the map. Suppose that each road has a *toll*, stored in $t(u, v)$, which represents the *cost* of taking the road between intersection u and v. Further more, a **tax** must be paid at certain positions on the map. This is stored in $u.tax$ which is a positive numerical value representing the tax at position u. Note that the tax value may be zero. Let s be a start position on the map and t be an end position on the map.

Your Job:. Determine the minimum cost in travelling from position s to position t. You must write the pseudo-code for your procedure, which outputs the minimum cost route. You must also provide the pseudo-code that prints out the best route. Justify the runtime of $O(m \log n)$.

(b) 10 points Consider a mountain biking trail map consisting of n trail markers, and m trails, where each trail connects two markers and travels in only ONE direct. Each trail also has an associated distance. Exactly 5 of the trail markers contain toilets. There is also a toilet at the START trail marker S and the FINISH trail marker F . A family would like to start hiking at marker S and hike to trail marker F . However, they would like to plan a route so that they never have to travel more than 20 miles without a bathroom break! Your job is to design an algorithm that finds the shortest overall route from S to F such that the condition is met. You must describe in detail the steps of your procedure, including what graph model you use, what attributes you use and what they are used for, and what algorithms (with input/output) you use from class. Justify the runtime of O(m log n).